**Deskripsi Judul Tugas Akhir**

**FINETUNING MODEL LAYOUTLM UNTUK PEMBACAAN NOTA BERBAHASA INDONESIA**



Oleh:

I Made Andre Dwi Winama Putra (1905551003)

Konsentrasi:

Teknologi Cerdas

**PROGRAM STUDI TEKNOLOGI INFORMASI**

**FAKULTAS TEKNIK**

**UNIVERSITAS UDAYANA**

**2022**

# PENDAHULUAN

Bab I pendahuluan pada laporan aplikasi OCR penyimpan bukti pembayaran ini berisi tentang latar belakang, rumusan masalah, batasan masalah, manfaat, serta sistematika yang akan digunakan dalam pembuatan laporan aplikasi OCR penyimpan bukti pembayaran ini.

## Latar Belakang

Transaksi keuangan merupakan sebuah interaksi yang biasa terjadi dalam hidup manusia. Kebutuhan untuk saling bertukar barang merupakan hal yang tidak dapat dihindari sebagai manusia yang merupakan makhluk sosial. Pencatatan dan pembukuan dari transaksi keuangan merupakan hal yang sangat penting untuk dilakukan oleh sebuah perusahaan ataupun organisasi yang sah, agar segala arus uang yang masuk dan keluar dapat diamati.

Tanda bukti dalam sebuah transaksi yang biasa ditemui saat ini adalah Struk, Nota dan Kuitansi. Bukti-bukti transaksi ini biasa dicetak didalam kertas yang membuat bukti-bukti ini mudah hilang. Selain itu, pemindahan informasi transaksi ke dalam bentuk digital tentunya akan memakan banyak waktu karena setiap data transaksi tersebut harus di-*input­* secara manual ke dalam komputer (Mohammad, Anarase, Shingote, & Ghanwat, 2014).

Keberadaan sistem yang dapat mengekstrak informasi yang terdapat dalam bukti pembayaran dan menyimpannya dalam format digital secara otomatis, tentunya akan dapat meningkatkan efisiensi kerja. Pemindahan media penyimpanan ke dalam bentuk digital juga akan mengurangi resiko bukti transaksi tersebut menghilang.

Ekstraksi informasi dengan menggunakan *Optical Character Recognition* (OCR) saat ini sudah memiliki akurasi yang cukup tinggi, baik dalam pengenalan tulisan digital (*digital character*) maupun dalam pengenalan tulisan tangan (*handwritten character*). Dengan keberadaan teknologi OCR, proses ekstraksi informasi yang terdapat pada bukti pembayaran struk dan nota akan dapat dilakukan secara otomatis (V. Kumar, P. Kaware, P. Singh, R. Sonkusare and S. Kumar. 2020).

*Optical Character Recognition* (OCR) adalah proses konversi gambar huruf menjadi karakter ASCII yang dikenali oleh komputer. Gambar huruf ini dapat berasal dari *scan* dokumen, tulisan digital, maupun tulisan tangan. Gambar yang terbaca oleh sistem OCR ini nantinya akan dikonversi menjadi format yang dapat dipahami mesin (Mohammad, Anarase, Shingote, & Ghanwat, 2014).

Keberadaan sebuah aplikasi OCR yang dapat menyimpan bukti pembayaran, pengguna aplikasi akan dapat menyimpan struk bukti pembayaran yang biasa berbentuk kertas menjadi format digital yang akan mengurangi resiko terjadinya kehilangan. Manfaat lainnya dari keberadaan aplikasi penyimpanan bukti pembayaran ini yaitu dapat melihat total pengeluaran mereka dalam jangka waktu tertentu tanpa perlu menghitungnya secara manual.

Sistem pembacaan bukti pembayaran ini sudah pernah dibuat dengan menggunakan metode *template matching* serta OCR.siapa yang buat Namun, dengan penggunaan metode *template matching*, pembacaan bukti pembayaran tidak dapat diimplementasikan pada bentuk nota pembayaran yang beragam, dikarenakan *template* nota yang berbeda.

Sistem yang dibangun akan bekerja dengan cara mengakuisisi citra bukti pembayaran yang diambil menggunakan kamera dari *smartphone* android terlebih dahulu. Citra tersebut selanjutnya akan dikirim menuju *web server* dengan menggunakan *API call.* *Web server* akan menerima dan memproses citra bukti tersebut hingga menjadi *text* dan akan dikirim kembali kepada *smartphone* android. Informasi tentang bukti pembayaran tersebut selanjutnya akan ditampilkan dalam layar *smartphone* dan pengguna dapat memilih untuk menyimpan bukti pembayaran tersebut.

## Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, maka masalah-masalah yang dapat dirumuskan adalah sebagai berikut.

1. Bagaimana cara merancang dan membangun aplikasi yang dapat membaca bukti pembayaran berbentuk kertas menjadi bentuk digital secara otomatis.
2. Bagaimanakah akurasi dari sistem pembacaan nota belanja yang dibuat.

## Tujuan Penelitian

Berdasarkan rumusan masalah yang didapatkan, tujuan yang ingin dicapai dari penelitian tugas akhir ini adalah sebagai berikut.

1. Menghasilkan sebuah aplikasi yang dapat membaca dan menyimpan bukti pembayaran dalam bentuk digital secara otomatis.

2. Menghasilkan sistem yang dapat mengekstrak informasi dalam bukti pembayaran dengan akurasi yang baik.

## Manfaat Penulisan

Manfaat penelitian ini dibuat berdasarkan masalah yang ingin diselesaikan, serta tujuan ingin dicapai oleh penulis. Penelitian ini diharapkan akan memberikan manfaat kepada Pengguna dari aplikasi, yaitu Pengguna dapat merasakan manfaat dari kemudahan pencatatan pembayaran hanya dengan menggunakan gambar serta dapat memahami dan melakukan manajemen keuangan dengan lebih baik.

## Batasan Masalah

Batasan masalah yang digunakan dalam penelitian ini dibuat untuk memastikan agar ruang lingkup penelitian tidak terlampau jauh dan melebar. Batasan masalah dari penyusunan laporan tugas akhir ini adalah sebagai berikut.

1. Dataset yang digunakan berasal dari berbagai nota belanja dari minimarket dan restaurant yang berupa nota *print out*.
2. Nota dapat terbaca dengan jelas.
3. Nota yang dideteksi berbentuk persegi panjang dan tidak terlipat.
4. Sistem hanya akan membaca satu buah nota pada setiap gambar yang dikirimkan ke *web server*.

## Sistematika Penulisan

Sistematika penulisan yang digunakan pada laporan tugas akhir ini terdiri dari pendahuluan, tinjauan pustaka, pembahasan metodologi, analisis, serta simpulan dan saran yang dirangkum secara urut dan sistematis.

### Bab I Pendahuluan

Bab ini dimulai dari penjelasan mengenai latar belakang diambilnya topik penelitian, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, hingga sistematika penulisan yang digunakan untuk menyusun penelitian ini.

### Bab II Tinjauan Pustaka

Bab ini berisikan tentang dasar-dasar teori yang bersumber dari artikel-artikel akademik, serta platform pembelajaran akademik yang digunakan sebagai dasar dalam pembahasan permasalahan dan solusi penelitian yang dilakukan.

### Bab III Metodologi Penelitian

Bab ini memuat informasi mengenai tempat dan waktu pelaksanaan penelitian, sumber dari data yang digunakan, serta analisis permasalahan yang akan diselesaikan.

### Bab IV Pembahasan dan Analisis Hasil

Bab IV berisikan informasi tentang pembahasan hasil dari penelitian mulai dari hasil dari pengumpulan data yang dilakukan, proses pemodelan, hingga evaluasi hasil penelitian secara keseluruhan.

### Bab V Penutup

Bab ini berisikan mengenai simpulan dan saran yang dibuat selama dilaksanakannya penelitian ini. Simpulan mengacu pada hasil yang didapatkan selama pengerjaan serta analisis hasil praktikum. Saran berisikan tentang masukan yang dapat diberikan kepada pembaca untuk mendapatkan hasil yang lebih baik, maupun untuk mempermudah replikasi dari penelitian.

# TINJAUAN PUSTAKA

Bab II ini berisikan tentang dasar-dasar teori yang bersumber dari artikel-artikel akademik, serta platform pembelajaran akademik yang digunakan sebagai dasar dalam pembuatan Aplikasi Pembacaan Struk Belanja ini.

## State of The Art

*State of The Art* adalah artikel dan literatur yang membahas topik serupa dengan penelitian ini. Pembuatan *State of The Art* dilakukan dengan tujuan untuk membandingkan teknologi yang saat ini digunakan untuk menyelesaikan masalah yang serupa, yaitu tentang pengenalan *Natural Language Processing* (NLP) dengan menggunakan *Optical Character Recognition* (OCR).

#CNN boleh untuk pembacaan karakter boleh masuk cari yang menggabungkan kata. Ga harus sampai baca suatu dokumen

Dalam artikel (Lin et al., 2022) yang berjudul “*Automatic Receipt Recognition System Based on Artificial Intelligence Technology*“, pembacaan OCR pada nota belanja dilakukan dengan mengaplikasikan *template matching* pada nota yang ingin dibaca. Hasil dari *template matching* tersebut selanjutnya akan dimasukkan ke dalam model *deep learning Yolov4-s*. Sistem yang dikembangkan ini berhasil mendapatkan akurasi sebesar 80.93% dengan percobaan menggunakan CNN dan sebesar 99.39% dengan sistem *Yolov4-s* yang dikembangkan dalam pembacaan karakter.

Dalam artikel (Andreas et al., 2020) yang berjudul “*Pembangunan Aplikasi Tracking Pemasukan dan Pengeluaran Keuangan Memanfaatkan Teknologi OCR BlinkReceipt Mobile SDK*“, pembacaan nota dilakukan dengan memanfaatkan *library* *BlinkReceipt*. *BlinkReceipt* merupakan sebuah *Application Programming Interface* (API)yang dapat digunakan dalam platform *IOS*, *Android*, serta melalui *javascript* yang merupakan *library* untuk membaca nota belanja. Hasil yang didapatkan memiliki beberapa kekurangan seperti aplikasi tidak dapat membaca potongan harga, serta pembacaan harus dilakukan berulang-ulang jika jumlah daftar belanja dalam struk cukup banyak.

Dalam artikel (Raoui-Outach et al., 2018). yang berjudul “*Deep Learning for automatic sale receipt understanding*“, pembacaan nota yang dilakukan dengan menggunakan lokalisasi gambar nota belanja dengan cara deteksi tepi yang digunakan untuk mengklasifikasi merek toko serta pembacaan OCR dengan menggunakan *Deep Convolutional Neural Networks* hasil dari percobaan ini berhasil melokalisasi merek toko sebesar 86%.

Dalam artikel (Puspitarani & Syukriyah, 2017) yang berjudul “*Pemanfaatan Optical Character Recognition Dan Text Feature Extraction Untuk Membangun Basisdata Pengaduan Tenaga Kerja*”, pembacaan pengaduan ketenagakerjaan dilakukan menggunakan *Tesseract OCR* dan *NTLK toolkit.* Hasil dari penelitian ini berhasil mengekstrak informasi sebesar 66.7% pada surat hasil OCR dan sebesar 91,67% pada surat yang diketik secara manual.

Dalam artikel (Logan et al. 2022) yang berjudul “*Segmentation for document layout analysis: not dead yet*”, mengajukan metode *weighted bounding box regression loss* untuk meningkatkan akurasi dalam segmentasi dokumen yang memiliki banyak objek-objek kecil. Hasil yang didapatkan dalam pengujian *Dense Article Dataset* (DAD) dan PubLayNet dataset memiliki f1 *score* sebesar 96.26% and 97.11% dengan menggunakan model DeeplabV3+.

Dalam artikel (Duc-Dung Nguyen, 2021) yang berjudul “*TableSegNet:* *a fully convolutional network for table detection and segmentation in document images*”, mengajukan sebuah model baru untuk mendeteksi tabel dalam sebuah gambar dokumen. TableSegNet menggunakan arsitektur *fully convolutional network* untuk mendeteksi dan membedakan tabel secara bersamaan. Hasilnya model ini dapat menghasilkan akurasi sebesar 90% dalam dataset ICDAR2019.

Dalam artikel (Yichuan et al. 2021) yang berjudul “*MRZ code extraction from visa and passport documents using convolutional neural networks*”, sebuah model didesain untuk dapat mendeteksi *MRZ code* dari gambar Passport digital dengan menggunakan *Convolutional Neural Networks*. Hasilnya model ini dapat mendeteksi 100% dari kode MRZ dan 99.25% macro-f1 pada dataset Passport dan Visa.

Dalam artikel (Holeček, 2020) yang berjudul “*Learning from similarity and information extraction from structured documents*”, menggunakan arsitektur *Siamese networks* dengan kombinasi konsep *similarity*, *one-shot learning*, dan *context/memory awareness*, berhasil meningkatkan skor f1 sebesar 8.25% bila dibandingkan dengan arsitektur *Query answer.*

## Python

*Python* merupakan sebuah bahasa pemrograman tingkat tinggi yang bersifat *general purpose* dan mengutamakan kemudahan pembacaan dalam sintaks kodenya sehingga membuat bahasa *Python* menjadi bahasa pemrograman yang sangat populer. Saat ini bahasa *Python* masih terus dikembangkan dengan versi terakhirnya adalah versi 3.10.x. Bahasa pemrograman *Python* juga dapat digunakan dalam penulisan *Object Oriented Programming* seperti pada bahasa Java dan C++. Bahasa pemrograman *Python* memiliki lisensi *General Public Lisence* yang berarti siapapun dapat memakai dan mengembangkan bahasa ini. Kemudahan pembacaan sintaks serta lisensi yang bersifat *General Public Lisence* ini membuat bahasa ini sangat diminati dan saat ini telah men-*support* berbagai *library* yang membuat bahasa *Python* ini menjadi salah satu bahasa terbaik untuk melakukan *Data Science*.

## OpenCV

*Open* *Computer Vision* atau yang kerap disingkat menjadi *Opencv* merupakan sebuah *library* yang memuat berbagai *tools* untuk melakukan pengolahan pada gambar. Mulai dari pengambilan gambar, pemrosesan pada gambar seperti pengubahan dimensi warna gambar, filter warna pada gambar, deteksi tepi, penambahan elemen gambar, segmentasi, serta untuk menyimpan gambar. *Library Opencv* tersedia dan dapat digunakan dalam berbagai bahasa pemrograman, seperti *Python, C++, Java*, dll.

## SQLite

*SQLite* merupakan sebuah *database* tanpa *server* dan bersifat *embedded,* yang berarti *database* *SQLite* ini tertanam langsung pada aplikasi. Dalam segi tipe data *SQLite* hanya men-*support* tipe data *Blob, Integer, Null, Text,* dan *Real* berbeda dengan *database* *MySQL* yang men-*support* lebih banyak tipe data namun berjalan pada *server*. *Database* *SQLite* telah di-*support* oleh berbagai macam sistem, salah satunya adalah sistem operasi Android.

## OCR

*Optical Character Recognition* (OCR) adalah proses mengkonversi *text* yang terdapat di dalam citra gambar menjadi ke dalam format *text* yang dapat dibaca oleh mesin. Teknologi OCR ini merupakan bagian dari *Artificial Intelligence* yang banyak digunakan dalam bidang automasi seperti pemindaian dokuman, pembacaan plat nomor kendaraan, verifikasi dokumen, dll. Beberapa arsitektur model yang dapat digunakan sebagai dasar untuk melakukan OCR adalah model *Long Short Term Memory* (LSTM), *Convolutional Neural Network* (CNN), dll. Model OCR sering dipadukan dengan model *Natural Language Processing* (NLP) untuk meningkatkan akurasi dari pembacaan *text.* Salah satu arsitektur NLP yang sering digunakan dalam hal ini adalah *Bidirectional Encoder Representations from Transformers* (BERT).

## Flask

*Flask* adalah sebuah *framework* aplikasi *web* yang ditulis menggunakan bahasa pemrograman Python. *Framework* *Flask* ini berperan untuk mengatasi berbagai permasalahan pada sebuah *server*, melakukan *Object Relational Mapping* (ORM) pada *database*, serta memiliki berbagai macam ekstensi untuk mempermudah pembuatan *web*site dengan menggunakan bahasa *Python*. Kelebihan dari penggunaan *framework Flask* ini adalah *framework* ini tergolong ringan untuk digunakan. Selain itu, *framework Flask* juga bersifat *beginner friendly* serta sudah men-*support* pengolahan *HTTP request*, *database*, *autentikasi, mail* *service*, dan firut-fitur lainnya yang membuat *framework Flask* ini sesuai untuk digunakan dalam tahap pengembangan aplikasi.

## LayoutLM

Layoutlm merupakan sebuah model *document understanding* yang dibuat pada tahun 2019 oleh (Xu et al., 2020) dan terbit dalam artikel arXiv:1912.13318 yang berjudul “*LayoutLM: Pre-training of Text and Layout for Document Image Understanding*”. Model ini adalah sebuah model yang dibuat dengan memperhatikan perkembangan permasalahan *Natural Language Processing* (NLP), di mana pada setiap model NLP fokus utama selalu terdapat pada *text-level manipulation*. Model layoutlm ini dibuat dengan menggunakan interaksi antar informasi pada *teks* yang ada pada sebuah dokumen beserta *layout* dari dokumen tersebut. Pengembangan model ini dilakukan dengan menggunakan data dari dokumen-dokumen yang telah di-*scan­* yang berasal dari berbagai macam kategori seperti surat, *memo, email, invoice, news, articles, questionaire, resume*, dll.

|  |
| --- |
|  |

Gambar Arsitektur LayoutLM *(tambahin umber)*

Arsitektur *LayoutLM* ini menjelaskan alur kerja pemrosesan data dalam model *LayoutLM.* Arsitektur ini bekerja dengan cara membagi pemrosesan data menjadi dua jenis yaitu *text* serta posisi dari *text* tersebut. Pertama *pre-build* OCR *parser* akan digunakan untuk mengekstrak informasi *text* dan posisi *text* tersebut dalam *embedding*. Kedua informasi tersebut selanjutnya akan dilatih menggunakan Faster R-CNN, dan hasilnya diproses lebih lanjut pada *FC Layers*. *Output* dari arsitektur ini berupa *embedding* dari setiap informasi gambar yang diberikan.

# METODOLOGI PENELITIAN

Bab III merupakan metodologi penelitian yang menjelaskan tentang tempat dan waktu penelitian, gambaran umum sistem, alur aplikasi dan perancangan *database*.

## Tempat dan Waktu Penelitian

Pembuatan tugas akhir sistem pembacaan nota ini dilakukan di rumah penulis dan Kampus Teknologi Informasi Universitas Udayana. Waktu yang digunakan untuk melaksanakan penelitian dimulai pada bulan September 2021 sampai dengan bulan Desember 2022.

## Data Penelitian

Data pelatihan yang digunakan dalam penelitian ini menggunakan data sekunder yang ditambahkan dengan data primer. Data primer merupakan data yang dikumpulkan oleh penulis untuk melakukan penelitian ini. Sedangkan data sekunder merupakan data yang tersedia secara publik dan dapat diambil untuk membantu penelitian.

### Data Primer

Data Primer yang digunakan dalam *finetuning* model LayoutLM ini merupakan data yang dikumpulkan dari hasil bertransaksi pada berbagai supermarket, dan minimarket yang memberikan nota belanja *print out* digital. Total dari nota belanja yang telah dikumpulkan berjumlah //000. Nota-nota belanja ini nantinya akan difoto untuk dijadikan gambar yang digunakan sebagai data pelatihan model LayoutLM.

### Data Sekunder

Data Sekunder yang digunakan pada penelitian ini menggunakan dataset *wildreceipt* oleh Theivaprakasham yang merupakan dataset nota belanja yang telah discan serta memiliki informasi setiap kata, letak kata bounding box, serta label untuk setiap kata yang ada dalam nota belanja. Dataset *wildreceipt* ini memiliki data berjumlah 1267 data latih 472 data evaluasi.

## Gambaran Umum Sistem

Gambaran umum sistem menjelaskan tentang desain perancangan dari sistem yang akan digunakan pada pembuatan sistem pembacaan nota ini. Sistem yang dibuat akan terbagi menjadi rancangan server beserta rancangan aplikasi android sebagai berikut.

|  |
| --- |
|  |

Merupakan gambaran *flowchart* dari aplikasi android yang akan dibuat. Aplikasi ini memiliki dua fungsi utama yaitu untuk mengirim foto dari nota belanja kepada *web server* menggunakan *API Call* dan menyimpan hasil pembacaan nota yang telah diterima dari *web server* serta untuk melihat *history* data belanja bulanan yang tersimpan pada sistem.

|  |
| --- |
|  |

Merupakan gambaran *flowchart* dari *web server* yang dijalankan dengan menggunakan *framework Flask.* *Web server* akan menerima gambar yang telah dikirimkan dari aplikasi android dan melakukan *preprocessing* pada gambar agar gambar tersebut dapat terbaca pada sistem. Setelah itu, gambar akan diproses kembali untuk mendapatkan *embedding* text dan *layout* notaserta hasil prediksi model. Hasil prediksi tersebut kemudian akan diinterpretasikan menjadi data tanggal, barang belanja, subtotal item, serta total belanja dan dikirimkan kembali pada aplikasi android.

## Alur Aplikasi

Alur dari aplikasi pembacaan nota belanja ini dapat dilihat pada *use case diagrams* yang akan menjelaskan secara umum fungsi yang terdapat pada sistem sistem serta interaksi yang dapat dilakukan *user* dengan aplikasi.

|  |
| --- |
|  |

Use Case Diagram Aplikasi Pembacaan Nota Belanja

# Pakai preprocess di web server jangan dipisah

Terdapat dua buah fungsi utama dari aplikasi pembacaan nota belanja yang dapat dilihat pada gambar *use case diagrams* ini. Fungsi pertama adalah fungsi *scan* nota belanja yang dapat dilakukan dengan mengirimkan foto dari nota yang ingin dibaca kepada *web server*. Setelah itu, hasil dari pembacaan nota akan diberikan oleh *web server* untuk dapat disimpan pada sistem android. Fungsi kedua adalah untuk melihat data historis belanjaan yang telah tersimpan pada sistem.

## Data Preprocessing

Data yang digunakan dalam pengembangan sistem pembacaan nota belanja ini adalah data berupa gambar hasil scan dari nota serta gambar nota yang berupa foto. Preprocessing data diperlukan untuk menyamakan format dari kedua gambar tersebut.

### Segmentation

Segmentasi merupakan proses pemotongan atau pemilihan area *Region of Interest* (ROI) dari sebuah gambar untuk memisahkan objek dengan background. Segmentasi dilakukan pada gambar nota yang berupa foto untuk dapat menyatukan format gambar dengan gambar nota hasil scan.

### OCR Kata dan Layout

Nota belanja yang sudah disegmentasi akan dibaca dengan menggunakan proses OCR untuk mendapatkan kata-kata dan letak dari setiap kata yang ada dalam nota. Pembacaan kata dan layout ini dilakukan dengan menggunakan library Google Vision.

## Model Finetuning

Model finetuning adalah sebuah istilah dalam machine learning untuk menyelesaikan permasalahan *Natural Language Processing* (NLP). Finetuning adalah merupakan proses yang serupa seperti *transter learning* pada *Convolutional* model yang dapat menggunakan arsitektur yang sama untuk menyelesaikan berbagai permasalahan. Finetuning model LayoutLM dilakukan dengan melatih model dasar dengan data baru yaitu nota yang telah dikumpulkan.

|  |
| --- |
|  |

Flowchart Pelatihan LayoutLM

Gambar // merupakan *flowchart* dari proses pelatihan model LayoutLM. Pertama model base LayoutLM akan dimuat kedalam sistem. Setelah itu akan dilakukan proses normalisasi bounding box pada setiap kata dalam nota yang telah discan oleh Google Vision. Setiap kata dan bounding box pada nota akan dimuat kedalam proses *auto encoder* yang dimiliki oleh Model LayoutLM untuk menyamakan format data dengan format original pelatihan. Terakhir Model akan mengalami proses pelatihan.

## Evaluasi Model

Evaluasi model *finetuned* LayoutLM dilakukan dengan cara membandingkan hasil prediksi label dari setiap kata, dengan label aslinya. Evaluasi model akan dilakukan dengan menggunakan *confussion matrix* serta *classification report* yang akan menunjukan hasil pada setiap label yang berbeda.

### Confussion Matrix

Evaluasi dengan menggunakan confussion matrix dapat menunjukan hasil prediksi label dari setiap kelas dengan lebih baik. Dengan melihat kombinasi nilai *true positive* (TP), *false positive* (FP), *false negatif* (FN)dan *true negative* (TN) dari setiap label, dapat ditentukan apakah model sudah belatih dengan baik.

### Classification Report

Evaluasi yang dilakukan dengan menggunakan *classification report* menunjukan dengan lebih baik akurasi, presisi, dan recall dari model yang telah dibuat. Evaluasi dengan menggunakan *classification report* juga menunjukan nilai akurasi, presisi, dan recall pada setiap label yang ada dalam proses pelatihan.

## Model Deployment

*Model* deployment adalah proses pemindahan model ke tahap produksi untuk dapat digunakan dalam memprediksi data baru. Proses model *deployment* dilakukan dengan meyimpan model dalam bentuk //. Setelah itu, sebuah webserver akan dibuat dengan *library* Flask untuk memuat model dan melakukan proses *inferensi* secara real-time dengan menggunakan API Call untuk mengirimkan gambar pada server.

# HASIL DAN PEMBAHASAN

Bab IV Hasil dan Pembahasan membahas mengenai proses pengerjaan sistem yang digunakan dalam memalukan *finetuning model* LayoutLM untuk membaca nota berbahasa indonesia beserta hasil yang ditemukan selama pengerjaan sistem.

## Pembuatan Dataset

Pembuatan dataset dilakukan dengan cara mengumpulkan nota belanja yang didapatkan setelah bertransaksi di toko Alfamart, Circle K, Indomaret, Mixue, dll. Nota-nota tersebut selanjutnya akan difoto menggunakan perangkat *smartphone* Samsung //.

### Segmentation

Proses segmentasi dilakukan pada gambar nota yang telah dikumpulkan untuk menghilangkan background dan mendapatkan gambar yang berfokus kepada nota yang merupakan informasi utama dalam pengerjaan sistem

|  |
| --- |
| file\_name = '20221013\_192759.jpg'  file\_path = os.path.join(DIR,'Nota',file\_name)  img\_read = cv2.imread(file\_path)  img\_read = cv2.cvtColor(img\_read, cv2.COLOR\_BGR2RGB)  if img\_read is None:  raise Exception(f"Image {file\_name} not found")    resize\_ratio = 1000 / img\_read.shape[0]  img\_rezise = resize\_img(img\_read, resize\_ratio)  gray = cv2.cvtColor(img\_rezise, cv2.COLOR\_BGR2GRAY)  plt.imshow(img\_rezise)  plt.show() |

S

Kode Program 4.1 merupakan kode program yang digunakan untuk membaca file foto dengan menggunakan *library* OpenCv. Pertama dilakukan proses resize\_img untuk memperkecil ukuran gambar dan operasi cv2.cvtColor untuk mengambil gambar dalam bentuk grayscale

|  |
| --- |
|  |

Gambar 4.1

Gambar 4.1 adalah hasil dari proses pembacaan gambar dan proses resize\_img yang telah dilakukan. Gambar nota hasil akan memiliki ukuran panjang 1000. Selain itu sebuah gambar *grayscale* akan disimpan dalam variabel gray.

|  |
| --- |
| blurred = cv2.GaussianBlur(gray, (3, 3), 3)  rectKernel = cv2.getStructuringElement(cv2.MORPH\_RECT, (5, 5))  dilated = cv2.dilate(blurred, rectKernel)  plt.figure(figsize=(10,10))  plt.imshow(dilated, cmap='gray')  plt.show() |

Kode Program

Kode Program 4.2 adalah kode program untuk memproses gambar *grayscale* dengan menggunakan proses *blurring*, dan proses dilasi. Proses *blurring* dilakukan dengan menggunakan GaussianBlur pada gambar *grayscale* dan proses dilate pada gambar yang telah di-*blur*.

|  |
| --- |
|  |

G

Gambar 4.2 adalah hasil dari proses *blurring* dan dilasi pada gambar *grayscale*. Proses *blurring* dan proses dilasi dilakukan pada gambar agar tulisan pada gambar menjadi tidak terbaca saat dilakukan deteksi tepi.

|  |
| --- |
| def auto\_canny(image, sigma=1):  # compute the median of the single channel pixel intensities  v = np.median(image)  # apply automatic Canny edge detection using the computed median  lower = int(max(0, (1.0 - sigma) \* v))  upper = int(min(255, (1.0 + sigma) \* v))  edged = cv2.Canny(image, lower, upper)  # return the edged image  return edged  edged = auto\_canny(dilated)  plt.figure(figsize=(10,10))  plt.imshow(edged)  plt.show() |

|  |
| --- |
|  |

|  |
| --- |
| detected\_lines = cv2.HoughLinesP(edged,rho = 0.5,theta = 1\*np.pi/180,threshold = 30,minLineLength = 20,maxLineGap = 100)  edged\_line = np.zeros\_like(edged)  line\_extender = 0  for line in detected\_lines:  x1, y1, x2, y2 = line[0]  is\_vertical = abs(x1 - x2) < abs(y1 - y2)  if is\_vertical:  x1 = int(x1-(abs(x1 - x2)/2\*line\_extender)\*0.314)  x2 = int(x2+(abs(x1 - x2)/2\*line\_extender)\*0.314)  if y1<y2:  y1 = int(y1-(abs(y1 - y2)/2\*line\_extender))  y2 = int(y2+(abs(y1 - y2)/2\*line\_extender))  else:  y1 = int(y1+(abs(y1 - y2)/2\*line\_extender))  y2 = int(y2-(abs(y1 - y2)/2\*line\_extender))  else:  x1 = int(x1-(abs(x1 - x2)/2\*line\_extender))  x2 = int(x2+(abs(x1 - x2)/2\*line\_extender))  y1 = int(y1-(abs(y1 - y2)/2\*line\_extender)\*0.314)  y2 = int(y2+(abs(y1 - y2)/2\*line\_extender)\*0.314)    cv2.line(edged\_line, (x1, y1), (x2, y2), (255), 1)  #main and backup method  contours\_m, \_ = cv2.findContours(edged, cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)  contours\_b, hierarchy = cv2.findContours(edged\_line, cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)  all\_conturs\_m = cv2.drawContours(img\_rezise.copy(), contours\_m, -1, (0,255,0), 3)  all\_conturs\_b = cv2.drawContours(img\_rezise.copy(), contours\_b, -1, (0,255,0), 3)  plt.figure(figsize=(15,15))  plt.subplot(1,2,1)  plt.imshow(all\_conturs\_m)  plt.subplot(1,2,2)  plt.imshow(all\_conturs\_b)  plt.show() |

|  |
| --- |
|  |

|  |
| --- |
| largest\_contours = sorted(contours\_m, key = cv2.contourArea, reverse = True)[0:4]  image\_with\_largest\_contours = cv2.drawContours(img\_rezise.copy(), largest\_contours, -1, (0,255,0), 3)  plt.figure(figsize=(10,10))  plt.imshow(image\_with\_largest\_contours)  plt.show() |

|  |
| --- |
|  |

|  |
| --- |
| try:  receipt\_contour = get\_receipt\_contour(largest\_contours)  detected\_receipt = cv2.drawContours(img\_rezise.copy(), [receipt\_contour], -1, (0, 255, 0), 2)  result = wrap\_perspective(img\_rezise.copy(), contour\_to\_rect(receipt\_contour,corner\_tolerance = 0))  print("using main method successfully")  except:  print("using backup method")  longest = 0  for i,cont in enumerate(largest\_contours):  x,y,w,h = cv2.boundingRect(largest\_contours[i])  cur\_length = w+h  if (cur\_length>longest):  longest = cur\_length  biggest\_idx = i  x,y,w,h = cv2.boundingRect(largest\_contours[biggest\_idx])  result = img\_rezise[y:y+h, x:x+w]  plt.figure(figsize=(10,10))  plt.imshow(result)  plt.show() |

|  |
| --- |
|  |

### Pembacaan Karakter

|  |
| --- |
| ocr\_agent = lp.GCVAgent.with\_credential(os.path.join(DIR,'gcv\_credential.json'),  languages = ['id'])  ocr\_agent  res = ocr\_agent.detect(image\_result, return\_response=True)  texts = ocr\_agent.gather\_text\_annotations(res)  lp.draw\_text(image\_result, texts, font\_size=12, with\_box\_on\_text=True,  text\_box\_width=3) |

|  |
| --- |
|  |

|  |
| --- |
| import json  from tqdm.notebook import tqdm  receipt\_list = {}  ocr\_agent = lp.GCVAgent.with\_credential(os.path.join(DIR,'gcv\_credential.json'), languages = ['id'])  for filename in tqdm(os.listdir('Nota\_Segmented')):  filepath = os.path.join(DIR,'Nota\_Segmented',filename)  image\_result = get\_receipt(filepath)  res = ocr\_agent.detect(image\_result, return\_response=True)  texts = ocr\_agent.gather\_text\_annotations(res)    inference\_words = []  for words\_bbox in texts:  inference\_words.append(words\_bbox.text)  inference\_boxes = []  for words\_bbox in texts:  h = np.min(words\_bbox.block.points, axis=0)  w = np.max(words\_bbox.block.points, axis=0)  inference\_boxes.append([h[0],h[1],w[0],w[1]])  receipt\_json = {}  receipt\_json['file\_name'] = filename  receipt\_json['size'] = image\_result.shape  receipt\_json['bboxes'] = [normalize\_bbox(box, image\_result.shape) for box in inference\_boxes]  receipt\_json['words'] = inference\_words    receipt\_list[filename] = receipt\_json  with open(f"Annotation/{filename.split('.jpg')[0]}.json", "w") as outfile:  json.dump(receipt\_list, outfile) |

### Annotasi Dataset

|  |
| --- |
| def annot\_helper(file\_name,bboxes,words,batch\_num):  file\_name = file\_name.split('.json')[0]+'.jpg'  img = cv2.imread(os.path.join(DIR,'Nota\_Segmented',file\_name))  img = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)  ex\_box = []  for bbox in bboxes:  bbox = unnormalize\_bbox(bbox, img.shape)  ex\_box.append(bbox)  ex\_word = words[batch\_num\*10:(batch\_num+1)\*10]  ex\_box = ex\_box[batch\_num\*10:(batch\_num+1)\*10]  for box in ex\_box:  cv2.rectangle(img, (box[0], box[1]), (box[2], box[3]), (255, 0, 0), 1)  plt.figure(figsize=(10,10))  plt.imshow(img)  plt.figure(figsize=(20,10))  for i in range(len(ex\_word)):  plt.subplot(2,5,i+1)  plt.imshow(img[ex\_box[i][1]:ex\_box[i][3], ex\_box[i][0]:ex\_box[i][2]])  plt.title(ex\_word[i])  plt.show()  file\_name = '20221013\_192759.json'  with open(os.path.join(DIR,'Annotation',file\_name)) as f:  data = json.load(f)    annot\_helper(file\_name,data['bboxes'],data['words'],batch\_num=4) |

|  |
| --- |
|  |

## Finetuning Model LayoutLM

### Data Preparation

### Data Pipelining

### Model Finetuning

### Model Evaluation

## Deployment Model LayoutLM

### Flask API Server

### Android APK

#### Pembacaan Nota

#### Review Pembacaan Nota

#### Penyimpanan Hasil Pembacaan Pada SQLite

**Referensi**

Bhadra, S., Aneja, K., & Mandal, S. (2022). OCR Using Convolution Neural Network in *Python* with Keras and TensorFlow. *International Journal of Advanced Research in Science, Communication and Technology*, *2*(1), 285–292. <https://doi.org/10.48175/ijarsct-2283>

Kumar, V., Kaware, P., Singh, P., Sonkusare, R., & Kumar, S. (2020). Extraction of information from bill receipts using optical character recognition. *Proceedings-International Conference on Smart Electronics and Communication, ICOSEC 2020*, *Icosec*, 72–77. <https://doi.org/10.1109/ICOSEC49089.2020.9215246>

Khandokar, I., Hasan, M., Ernawan, F., Islam, S., & Kabir, M. N. (2021). Handwritten character recognition using convolutional neural network. *Journal of Physics: Conference Series*, *1918*(4). <https://doi.org/10.1088/1742-6596/1918/4/042152>

Saw Zay Maung Maung. (2019). Convolutional Neural Network based Recognition of Myanmar *Text* Warning Sign for Mobile Platform. *International Journal of Engineering Research And*, *V8*(01), 216–221. <https://doi.org/10.17577/ijertv8is010102>

F. Mohammad, J. Anarase, M. Shingote, P. Ghanwat, 2014, Optical character recognition implementation using pattern matching, International Journal of Computer Science and Information Technologies 5, no. 2, pp: 2088-2090, 2014.

Maslova, O., Klein, L., Dabernat, D., Benoit, A., & Lambert, P. (2019). Receipt automatic reader. Proceedings-International Workshop on Content-Based Multimedia Indexing, 2019-Septe. https://doi.org/10.1109/CBMI.2019.8877407

Lin, C. J., Liu, Y. C., & Lee, C. L. (2022). Automatic Receipt Recognition System Based on Artificial Intelligence Technology. Applied Sciences (Switzerland), 12(2). https://doi.org/10.3390/app12020853

Raoui-Outach, R., Million-Rousseau, C., Benoit, A., & Lambert, P. (2018). Deep learning for automatic sale receipt understanding. Proceedings of the 7th International Conference on Image Processing Theory, Tools and Applications, IPTA 2017, 2018-Janua, 1–6. https://doi.org/10.1109/IPTA.2017.8310088

Rismanto, R., Prasetyo, A., & Irawati, D. A. (2020). Optimalisasi Image Thresholding pada Optical Character Recognition Pada Sistem Digitalisasi dan Pencarian Dokumen. Petir, 13(1), 1–11. <https://doi.org/10.33322/petir.v13i1.659>

Andreas, Y., Gunadi, K., & Purbowo, A. N. (2020). Implementasi Tesseract OCR untuk Pembuatan Aplikasi Pengenalan Nota pada Android. Artikel Infra, 8(1), 2–7.

Ref 2

<https://doi.org/10.1007/s10032-021-00390-4>

<https://doi.org/10.1007/s10032-021-00384-2>

<https://doi.org/10.1007/s10032-021-00375-3>